Amendments to the Claims

A complete list of pending claims follows, with indicated amendments:

1. (Currently Amended) A storage medium containing software for manipulating computer-implemented objects in a distributed system, the software comprising:

code to create a shared environment, the shared environment comprising an object-oriented programming environment distributed across multiple computer systems and comprising a plurality of objects; and

code to create an object, the object exposed to other objects in the shared environment, the object comprising:

a set of behavior logics, each member of the set of behavior logics operable to cause the object to perform a task; a receiver logic operable to receive a command from another object in the shared environment, wherein the receiver logic is externally invokable; and

a mapping logic able to map a command received at the receiver logic, on the basis of a characteristic of the command, to a selected behavior logic for execution of the selected behavior logic, wherein the mapping logic is within the object and wherein the existence of the mapping logic within the object allows the object to function as an autonomous unit such that the object can be moved within the computer systems of the shared environment and function independently of its location in the shared environment without the necessity of defining relationships between the object and other objects of the shared environment.

2. (Previously Amended) The storage medium of claim 1, wherein the set of behavior logics and the mapping logic are private to the object.

3. (Previously Amended) The storage medium of claim 1, wherein the set of behavior logics has no members.

4. (Previously Amended) The storage medium of claim 1, the object further comprising:

a default behavior logic operable to cause the object to perform a default task, wherein the default behavior logic is private to the object and wherein the default behavior logic is executed if the received command is not mapped to another behavior logic.

5. (Previously Amended) The storage medium of claim 1, wherein a command can be mapped to multiple behavior logics.

6. (Previously Amended) The storage medium of claim 1, the object further comprising:

an authentication data, the authentication data providable to other objects for authenticating commands received from the other objects by the code to receive the command.

7. (Previously Amended) The storage medium of claim 6, wherein the command comprises the authentication data, and wherein the mapping of a command to a behavior logic may be restricted in response to the authentication data.

8. (Currently Amended) The storage medium of claim 1, the software further comprising:

code to create a first shadow ~~Shadow~~ of the object, the first shadow ~~Shadow~~ of the object operable to communicate with the object, the first shadow ~~Shadow~~ of the object being informed of changes to the object and the object being informed of changes to the first shadow ~~Shadow~~ of the object.

9. (Currently Amended) The storage medium of claim 8, wherein the first shadow ~~Shadow~~ of the object is a copy of the object.

10. (Currently Amended) The storage medium of claim 8, wherein the mapping of commands to behavior logics of the first shadow ~~Shadow~~ of the object differs from the mapping of commands to behavior logics of the object.

11. (Currently Amended) The storage medium of claim 8, the software further comprising:

code to create a plurality of shadows ~~Shadows~~ of the object operable to communicate with the object and the first shadow ~~Shadow~~ of the object, the object and the first shadow ~~Shadow~~ of the object being informed of changes to any of the plurality of shadows ~~Shadows~~ of the object and each of the plurality of shadows ~~Shadows~~ of the object being informed of changes to the object and changes to the first shadow ~~Shadow~~ of the object.

12. (Currently Amended) The storage medium of claim 8, the software further comprising:

code to promote the first shadow ~~Shadow~~ of the object into a new object.

13. (Currently Amended) The storage medium of claim 12, the software further comprising:

code to create a plurality of shadows ~~Shadows~~ of the object,

wherein executing the code to promote the first shadow ~~Shadow~~ of the object into a new object converts each of the plurality of shadows ~~Shadows~~ of the object into a shadow ~~Shadow~~ of the new object.

14. (Currently Amended) The storage medium of claim 12, the shared environment further comprising:

a plurality of computer systems, the object on a first computer system of the plurality of computer systems, the first shadow ~~Shadow~~ of the object on a second computer system ~~r~~ of the plurality of computer systems; and

code to manage the plurality of computer systems, executing the code to promote the first shadow ~~Shadow~~ of the object to a new object if the first computer system experiences a predetermined condition.

15. (Currently Amended) The storage medium of claim 1, the set of behavior ~~Behavior~~ logics further comprising:

code to modify the mapping logic to modify the mapping of commands to behavior logics.

16. (Previously Amended) The storage medium of claim 1,

wherein the shared environment is operable to execute on a plurality of computer systems; and

wherein the object has a location on one of the plurality of computer systems and wherein the object acts independently of its location.

17. (Previously Amended) The storage medium of claim 1, the object further comprising:

code to configure the mapping logic from an external data source.

18. (Previously Amended) The storage medium of claim 1, wherein the software is capable of using any networking protocol.

19. (Currently Amended) A method of manipulating a computer-implemented object in a distributed system, the method comprising the steps of:

creating a shared environment, the shared environment comprising an object-oriented programming environment distributed across multiple computer systems and comprising a plurality of objects; and

creating an object, wherein the object is exposed to other objects in the shared environment, and wherein the step of creating an object comprises the step of:

coding a set of behavior logics, each member of the set of behavior logics causing the object to perform a task;

manipulating the object, wherein the step of manipulating the object comprises the steps of:

receiving a command from another object of the plurality of objects in the shared environment;

selecting a behavior logic of the set of behavior logics corresponding to the command on the basis of a mapping logic within the object that maps commands to behavior logics of the set of behavior logics on the basis of a characteristic of the command, wherein the mapping logic is within the object and the existence of the mapping logic within the object allows the object to be autonomous and function within the shared environment without reference to the location of the object in the shared environment and without the necessity of defining relationships between the object and other objects in the shared environment; and

executing the selected behavior logic responsive to the command.

20. (Previously Amended) The method of claim 19, wherein the set of behavior logics and the mapping logic are private to the object.

21. (Previously Amended) The method of claim 19, further comprising the step of:

modifying the mapping logic to modify the mapping of commands to behavior logics.

22. (Previously Amended) The method of claim 19, the method further comprising the steps of:

coding a default behavior logic to cause the object to perform a default task;; and

executing the default behavior logic if no other behavior logic from the set of behavior logics is mapped to the received command.

23. (Previously Amended) The method of claim 19, wherein the set of behavior logics has no members.

24. (Previously Amended) The method of claim 19, wherein multiple behavior logics are mapped to and selected on the basis of a received command.

25. (Previously Amended) The method of claim 19, further comprising the step of:

creating an authentication data for the object.

26. (Previously Amended) The method of claim 25, the command comprising the authentication data, the method further comprising the step of:

restricting the mapping of commands to behavior logics in response to the authentication data.

27. (Currently Amended) The method of claim 19, further comprising the step of:

creating a first shadow ~~Shadow~~ of the object, the first shadow ~~Shadow~~ of the object operable to communicate with the object, the first shadow ~~Shadow~~ of the object being

informed of changes to the object and the object being informed of changes to the first shadow ~~Shadow~~ of the object.

28. (Currently Amended) The method of claim 27, the step of creating the first shadow ~~Shadow~~ of the object comprising the step of:

copying the object.

29. (Currently Amended) The method of claim 27, the step of creating the first shadow ~~Shadow~~ of the object comprising the step of:

modifying the mapping of commands to behavior logics of the first shadow ~~Shadow~~ of the object.

30. (Currently Amended) The method of claim 27, further comprising the step of:

creating a plurality of shadows ~~Shadows~~ of the object operable to communicate with the object and the first shadow ~~Shadow~~ of the object, the object and the first shadow ~~Shadow~~ of the object being informed of changes to any of the plurality of shadows ~~Shadows~~ of the object and each of the plurality of shadows ~~Shadows~~ of the object being informed of changes to the object and changes to the first shadow ~~Shadow~~ of the object.

31. (Currently Amended) The method of claim 27, further comprising the step of:

promoting the first shadow ~~Shadow~~ of the object into a new object.

32. (Currently Amended) The method of claim 31, further comprising the step of:

creating a plurality of shadows ~~Shadows~~ of the object,

converting each of the plurality of shadows ~~Shadows~~ of the object into a shadow ~~Shadow~~ of the new object, responsive to the step of promoting the first shadow ~~Shadow~~ of the object.

33. (Previously Amended) The method of claim 19, wherein the object has a location on a first computer system of the plurality of computer systems and wherein the object acts independently of its location.

34. (Original) The method of claim 19, the shared environment capable of using any networking protocol to communicate with another shared environment.

35. (Previously Amended) The method of claim 19, further comprising the step of:

creating the mapping logic from an external data source.

36. (Currently Amended) A method for developing an application for execution on at least one computer system from configurable objects having behavior logics capable of performing tasks, the method comprising the steps of:

defining within an object-oriented programming environment a plurality of objects, each object of the plurality of objects operable to receive and execute commands, each object exposed to each other object of the plurality of objects, the step of creating the plurality of objects comprising the steps of:

creating a set of behavior logics for an object;

mapping members of a first set of commands to members of the set of behavior logics, wherein the mapping function of an object is included within the object, wherein the existence of the mapping function within the object allows the object to be autonomous and function within the object-oriented programming environment without reference to the location of the object in the object-oriented programming environment and without the necessity of defining relationships between the object and other objects in the object-oriented programming environment;

mapping any command not a member of the first set of commands to a default behavior logic; and

configuring a receiver logic to receive a command and initiate the execution of a behavior logic corresponding to the command in response to the mapping of the command to the behavior logic.

37. (Currently Amended) The method of claim 36, further comprising the steps of:

creating a shadow ~~Shadow~~ of an object of the plurality of objects, the shadow ~~Shadow~~ configured such that sending a command to the shadow ~~Shadow~~ causes the object to act as if the command had been sent to the object.

38. (Prevoiusly Amended) The method of claim 37, each of the plurality of objects having a location on one of a plurality of computer systems, each of the plurality of objects being independent of the location of each other of the plurality of objects.

39. (Currently Amended) The method of claim 38, wherein a shadow ~~Shadow~~ of each of the plurality of objects is automatically created on each of the plurality of servers other than the server on which the object is located.

40. (Currently Amended) A processor-based system, comprising:

a first processor; and

a first storage device coupled to the first processor containing ~~a~~ software to manipulate computer-implemented objects in a shared environment, the software comprising:

code to create a shared environment, the shared environment comprising an object-oriented programming environment distributed across multiple computer systems and comprising a plurality of objects; and

code to create an object of the plurality of objects, the object exposed to other objects in the shared environment, the object comprising:

a set of behavior logics, each member of the set of behavior logics operable to cause the object to perform a task; and

a receiver logic operable to receive a command from another object in the shared environment, the receiver logic being externally invokable;

a mapping logic able to map a command received at the receiver logic to a selected behavior logic for execution of the selected behavior logic on the basis of a characteristic of the command, wherein the mapping logic is within the object and wherein the existence of the mapping logic within the object allows the object to function as an autonomous unit such that the object can be moved within the computer systems of the shared environment

and function independently of its location in the shared environment without the necessity of defining relationships between the object and other objects of the shared environment.

41. (Currently Amended) The processor-based system of claim 40, the object further comprising:

a default behavior logic operable to cause the object to perform a default task, wherein the default behavior ~~Behavior~~ logic is private to the object and wherein the default behavior logic is executed if the received command is not mapped to another behavior logic.

42. (Previously Amended) The processor-based system of claim 40, wherein a command can be mapped to multiple behavior logics.

43. (Peviously Amended) The processor-based system of claim 40, further comprising:

an input device coupled to the first processor,

wherein a first object of the plurality of objects is coupled to the input device such that manipulation of the input device sends a command from the first object to a second object of the plurality of objects without identifying the input device, the second object of the plurality of objects acting responsive to the command independent of the nature of the input device.

44. (Original) The processor-based system of claim 40, further comprising:

an output device coupled to the first processor,

wherein a first object of the plurality of objects is coupled to the input device such that a first object is capable of rendering a second object on the output device without identifying the output device to the second object.

45. (Currently Amended) The processor-based system of claim 40, further comprising:

a second processor;

a network, coupled to the first processor and the second processor;

a second storage device coupled to the second processor, the second storage device containing the software;

the software further comprising:

code to connect the shared environment to the network;

code to create a shadow ~~Shadow~~ on the second processor of the object on the first processor, the shadow ~~Shadow~~ and the object communicating with each other to inform the shadow ~~Shadow~~ of changes to the object and the object of changes to the shadow ~~Shadow~~.

46. (Prevoiusly Amended) A software architecture for manipulating computer-implemented objects on a plurality of computers, some of the plurality of computers having input devices and some of the plurality of computers having output devices, the software architecture implemented in an extensible object-oriented language, comprising:

a distributed system, comprising:

a plurality of shared environments, each of the plurality of shared environments comprising an object-oriented programming environment distributed across and

executing on a different computer of the plurality of computers, each of the plurality of shared environments comprising:

a CommandReceiver class, the CommandReceiver class comprising:

a set of Behavior private methods, each member of the set of Behavior methods operable to cause instantiations of the CommandReceiver class to perform a task; and

an executeCommand public method, operable to receive a Command from an object in the shared environment, the executeCommand public method comprising:

code to receive the Command;

code to select a Behavior private method of the set of Behavior private methods selected corresponding to a characteristic of the Command from a Command-Behavior mapping; and

code to execute the selected Behavior private method; and

a Kernel subclass of the CommandReceiver class, the Kernel class comprising:

code to instantiate objects of the CommandReceiver class;

code to destroy objects of the CommandReceiver class.

47. (Previously Amended) The software architecture of claim 46, further comprising:

a Pawn subclass of the CommandReceiver class, the Pawn subclass comprising:

code to register an instantiation of a Pawn with a Kernel object of the Kernel subclass;

code to determine whether an instantiation of the Pawn subclass is a real Pawn or a Shadow Pawn of a real Pawn, and

code to send State information about an instantiation of the Pawn subclass,

wherein Commands received by Shadow Pawns are sent to the real Pawn corresponding to the Shadow Pawn.

48. (Original) The software architecture of claim 46, further comprising:

a ControlDevice subclass of the CommandReceiver class corresponding to an input device for receiving input from the input device and sending Commands to other CommandReceiver objects.

49. (Original) The software architecture of claim 46, further comprising:

a Construct subclass of the CommandReceiver class corresponding to an output device for rendering objects of the CommandReceiver class with graphical attributes.

50. (Original) The software architecture of claim 46, further comprising:

a Console subclass of the CommandReceiver class for allowing a user of the distributed system to instantiate, modify, and destroy objects, and for allowing a user to send Commands to CommandReceiver objects.

51. (Original) The software architecture of claim 46, further comprising:

a Nengine subclass of the CommandReceiver class for serializing and deserializing CommandReceiver objects, transmitting and receiving the serialized CommandReceiver object across a network to a Nengine in another shared environment of the distributed system.

52. (Original) The software architecture of claim 51, further comprising:

a Node subclass of the CommandReceiver class, an instantiation of the Node subclass corresponding to a Pawn object for representing the Pawn object to a Nengine object for communicating State information corresponding to a Pawn to Shadow Pawns of the Pawn and for communicating Commands sent to a Shadow Pawn to the real Pawn corresponding to the Shadow Pawn.